

*Prolećni semestar, 2022/23*

*PREDMET:*

*SE325 - Upravljanje projektima razvoja softvera*

**Razvoj softvera za sistem pametna kuca**

Projektni zadatak

Profesor: **Nemanja Zdravković**

Asistent: **Tamara Vukadinovic** Student: **Uros Milovanovic** Indeks: **4191**

Niš, 2022

Sadržaj

[Uvod 3](file:///G:\uni\Treca%20godina\SE325\SE325-DZ01-UrosMilovanovic-4191.docx#_Toc1226760664)

[Apstrakt aplikacije 3](file:///G:\uni\Treca%20godina\SE325\SE325-DZ01-UrosMilovanovic-4191.docx#_Toc1306280551)

[Lista uspešnosti projekta (Project Success Chart) 4](file:///G:\uni\Treca%20godina\SE325\SE325-DZ01-UrosMilovanovic-4191.docx#_Toc2133821802)

[Teoretska postavka 4](file:///G:\uni\Treca%20godina\SE325\SE325-DZ01-UrosMilovanovic-4191.docx#_Toc2099554227)

[Primena liste uspešnosti projekta 4](file:///G:\uni\Treca%20godina\SE325\SE325-DZ01-UrosMilovanovic-4191.docx#_Toc870217261)

[Struktura poslova WBS (Work Breakdown Structure) 6](file:///G:\uni\Treca%20godina\SE325\SE325-DZ01-UrosMilovanovic-4191.docx#_Toc1467671605)

[Teorijska postavka 6](file:///G:\uni\Treca%20godina\SE325\SE325-DZ01-UrosMilovanovic-4191.docx#_Toc2051358390)

[Primena strukture poslova 6](file:///G:\uni\Treca%20godina\SE325\SE325-DZ01-UrosMilovanovic-4191.docx#_Toc665353249)

[Zaključak 7](file:///G:\uni\Treca%20godina\SE325\SE325-DZ01-UrosMilovanovic-4191.docx#_Toc1900620637)

# Uvod

U ovom dokumentu biće predstavljene tehnike i načini upravljanja projektima razvoja demonstrirane na aplikaciji pametna kuca. U pitanju je sistem koji će olakšati upravljanje kuce krajnjeg korisnika kao i sistem na kome će korisnik moći imati korisnički interfejs po dizajnu koji on izabere. Tehnike kojima će se pisati su:

* Lista uspešnosti projekta (Project Success Chart)
* Struktura poslova WBS (Work Breakdown Structure)

# Apstrakt aplikacije

Sistem se sastoji od sledecih komponenti. Aplikacija za tablet ili telefon preko koje ce se upravljati ceo sistem. On ce sadrzati sve informacije kao sto su: da li su vreata zakljucana, ukoliko je navijen alarm, da li su zavese zatvorene ili otvorene… Zatim imacemo komponente kao sto su zavesa, muzika – alarm, vrata, prozori. Isto jedna od bitnijih komponenti ce biti asistent koji ce da prepozna glas I uraditi odredjenu komandu ( zakljucati vrata, otvoriti prozor, zatvoriti zavesu). Sistem ce funkcionistati tako sto korisnik udje u aplikaciju koja prikupi sve podatke o komponentama I ponudi korisniku opcije. Na primer zavesa je otvorena korisnik ce imati opciju da je zatvori nece moci da je otvori jer je vec otvorena. Ili korisnik ce imati opciju da udje u kucu I pozove asistenta gde ce aplikacija prepoznati glas I u tom trenutku slusati koju komandu daje korisnik. U slucaju da ta komanda postoji aplikacija ce je izvrsiti u suprotnom ce obavestiti da ne postoji ta komanda.

# Lista uspešnosti projekta (Project Success Chart)

## Teoretska postavka

Lista uspešnosti projekta predstavlja jedan od sedam koraka na putu ka uspešno razvijenom projektu, pogotovo prilikom razvoja projekata malih projekata. Ovaj korak podrazumeva korišćenje grafikona, tabela, dijagrama i drugih načina vizualnih prikaza podataka, upravljanjem kojih na adekvatan način pratimo usepešnost razvoja jednog projekta. Ova metoda nam lista faktore prema kojima se meri uspeh ishoda jednog projekta.

Kako bismo adekvatno svaki od ovih faktora uzeli u obzir, prevashodno je od važnosti definisati cilj prijekta sa svim neophodnim funkcionalnostima koje proizvod koji razvijamo mora raspolagati, a prema kojima ćemo i voditi metriku uspešnosti samog projekta.   
 Konkretizacija rokova i ispunjenja zahteva u odnosu na vremesku odrednicu jeste od ključne važnosti prilikom planiranja uspešnog projekta – samim tim, rangiranje uspešnosti projekta direktno zavisi od očekivanja stejkholdera u odnosu na jedinicu vremena. Tada se u obzir uzimaju njihova očekivanja izrade funkcionalnosti i kvaliteta samog proizvoda, u odnosu na rokove. Zbog toga je važno da svi članovi tima u svakom trenutku budu informisani o očekivanjima stejkholdera koji su jedinstveni za svaki projekat, a koji neposredno utiču na merenje uspeha rukovođenja jednim projektom.

## Primena liste uspešnosti projekta

**Cilj projekta:**

Kreiranje mobilne aplikacije koja je služi za upravljanjem kuce i koja će da olakša upravljanje kuce korisnika na interaktivan način. Rok za iztadu projekta je 6 meseca, preciznije do kraja decembra 2022. godine.

**Isporuka delova softvera:**

1. Na početku projekta potrebno je nakon prikupljanja informacija i zahteva, koje dobijamo iz razgovora sa stejkholderima, da definišemo, analiziramo i validiramo te zahteve. Kao izlaz dobija se SRS dokument. Za ovaj početni deo određuje se da je potrebno 15 dana.
2. Sledeći korak, nakon adekvatno definisanih zahteva, jeste projektovanje arhitekture sistema za šta neće biti potrebno više od 5 dana.
3. Nakon što je projektovanje arhitekture završeno i odobreno, počinje se faza implementacije sistema, za koju je potrebno 25 radnih nedelja (okvirno 6 meseci). U ovoj fazi se vrši komplentna implementacija svih podsistema uz njihovo kontinualno testiranje i integraciju.
4. Završetkom faze inicijalizacije, sistem se nalazi u alfa fazi. Vrši se alfa testiranje sa predodređenom grupom test korisnika. Testiranje i analiza se vrši iterativno. Nakon faze testiranja koja traje nedelju dana narednih nedelju dana posle toga se vrši popravka potencijalnih defekata sistema. Ovaj proces se ponvalja najviše 3 puta ili dok se ne dostigne zadovoljavajući nivo rada sistema.

**Stejkholderi:**

1. Naručilac softvera
2. Korisnici sistema

**Kriterijumi uspešnosti:**

1. Projekat zadovoljava/ispunjuje sve zahteve sjektholdera. Svi zahtevi definisani u SRS dokumentu su ispunjeni u fleksibilnost neispunjavanja određenih nefunkcionalnih zahteva.
2. Projekat nije probio definisani budžet.
3. Veći broj korisnika sistema je zadovoljen korisničkim iskustvom.
4. Sistem je stabilan i ima brz odziv.
5. Naručilac softvera brzo pridobija nove korisnike na sistem.

# Struktura poslova WBS (Work Breakdown Structure)

## Teorijska postavka

Svakom projektu razvijanom na nivou tima bitna je organizacija zadataka koji se tiču razvoja samog proizvoda. Ovo podrazumeva da se rad na jednoj funkcionalnosti pojednostavljuje na veliki broj manjih zadataka od kojih se sačinjava lista zadataka. Svi jednostavniji zadaci jesu delovi veće funkcionalnosti za čiju je izradu predodređen rok ka stejkholderima, i koji treba estimirati i komunicirati transparentno, sa uračunatim rizikom i uz dodatno pridavanje pažnje izvornim zahtevima stejkholdera.

## Primena strukture poslova

Da bi isporučili naš sistem, potrebno je veće celine sistema podeliti na više manjih delova.

1. **Planiranje**
   1. Definisanje cilja
   2. Prikupljanje stejkholdera
   3. Prikupljanje zahteva
   4. Specifikacija zahteva
   5. Definisanje budžeta
   6. Formiranje tima
   7. Definisanje rokova
2. **Implementacija**
   1. Razvoj prototipa
   2. Razvoj arhitekture sistema
   3. Razvoj baze podataka
   4. Razvoj korisničkog interfejsa
   5. Razvoj funckionalnosti
3. **Testiranje**
   1. Testiranje funckionalnosti
   2. Testiranje korisničkog interfejsa
   3. Alfa testiranje
   4. Beta testiranje
4. **Validacija**
   1. Ispravljanje grešaka
5. **Produkcija**
   1. Isporuka sistema na tržište
6. **Održavanje**
   1. Anketiranje krajnjih korisnika
   2. Dodavanje novih zadataka
   3. Ispravljanje grešaka

# Lista zaduženja (Task Assignment List)

## Teoretska postavka

Raspodela odgovornosti ima veliki značaj u procesu razvoja softvera. Ovo podrazumeva konkretnu i transparentnu podelu poslova među članovima tima. Konkretnost i transparentnost kao dva stuba raspodele zadataka postoje radi preciznosti obavljanja posla, kao i međusobnoj informisanosti članova tima čija važnost dolazi do izražaja u trenucima spajanja jednostavnih zadataka u jednu opštu funkcionalnost samog proizvoda.

Vođa tima ima obavezu raspodele konkretnih zadataka pojedincima, i to u skladu sa njihovim znanjem, iskustvom i nivoom senioriteta. Važan uticaj na raspodelu poslova i listu zaduženja imaju i takozvani doušnici, odnosno posrednici u transferu znanja koje se tiče obavljanje samih projektnih zadataka. Ovi pojedinci zaduženi su za prenos korisnih informacija od strane razvoja drugog dela softvera koji može biti od pomoći onima koji razvijaju neki drugi segment proizvoda, kao i pružati neophodna pojašnjenja kojim se dobija na vremenu i efikasnosti projekta.

Veliki deo odgovornosti nose pojedinci koji listom zaduženja dobiju zadatak da odobre isporučeni deo proizvoda. Ponekad se ovaj zadatak ostavlja stejkholderima, medjutim, potrebno je uveriti se u kvalitet isporučenog segmenta proizvoda pre samog predstavljanja funkcionalnosti stejkholderima odnosno krajnjim korisnicima.

## Primena liste zaduženja (Tast assignment List)

Prilikom formiranja tima treba uzeti u obzir obim posla, budžet i rok projekta kako bi se formirao idealan tim. Na mom projektu tim čini 5 čoveka, definisani u tabeli ispod.

|  |  |
| --- | --- |
| Ime i prezime | Pozicija |
| Uroš Milovanović | Project Leader |
| Aleksa Cekić | Mobile developer |
| Milan Veličković | Frontend developer |
| David Stefanović | Backend developer |
| Uroš Milovanović | Designer |

**Lista zadataka koje potrebno izvršiti:**

1. Prikupljanje zahteva
2. Kreiranje arhitekture sistema
3. Kreiranje baze podataka
4. Dodeljivanje zadataka
5. Implementacija backend-a
6. Dizajn korisničkih interfejsa
7. Implementacija KI (Web)
8. Implementacija KI (Mobile)
9. Integraciono tesitranje
10. Ispravka grešaka
11. Implementacija bezbednostnih mehanizama
12. Postavljanje aplikacije na server

U sledećoj tabeli možemo videti kome je koj zadatak dodeljen, ko je podrška na zadatku, odakle se informacije za zadatak prikupljaju I ukoliko je potrebno ko pruža odobrenje za izvršavanje zadataka.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zadatak | Vlasnik zadatka | Podrška | Pružalac info. | Odobrenje |
| Prikupljanje zahteva | Uros Milovanović |  | Stejkholderi |  |
| Kreiranje arhitekture sistema | David Stefanović | Uros Milovanović |  |  |
| Kreiranje baze podataka | David Stefanović | Uros Milovanović |  |  |
| Dodeljivanje zadataka | Aleksa Cekić | David Stefanović |  |  |
| Implementacija backend-a | David Stefanović |  |  |  |
| Dizajn korisničkih interfejsa | Uroš Milovanović | Aleksa Cekić |  |  |
| Implementacija KI (Web) | Milan Veličković |  | Milan Veličković | Uroš Milovanović |
| Implementacija KI (Mobile) | Aleksa Cekić |  | Milan Veličković | Uroš Milovanović |
| Integraciono tesitranje | David Stefanović | Aleksa Cekić, Uroš Milovanović |  |  |
| Ispravka grešaka | Aleksa Cekić Stefan Gogić  David Stefanović |  |  |  |
| Implementacija bezbednostnih mehanizama | David Stefanović | Uros Milovanović |  |  |
| Postavljanje aplikacije na server | David Stefanović | Uros Milovanović |  |  |

# Lista rokova (Task Duration List)

## Teoretska postavka

Lista rokova jeste faktor koji u velikom delu utiče na vreme izrade funkcionalnosti na softveru, a u krajnjoj liniji, vreme izrade kompletnog proizvoda. Organizacijski gledano, ova lista mora do detalja ispratiti listu zahteva o kojoj smo govorili u prethodnom pasusu. Kako bi se izbeglo nerealno ubrzavanje ili usporavanje samog razvoja, važno je realno estimirati odnosno predvide ti potrebno vreme izrade svake od funkcionalnosti. U velikoj meri će posao olakšati lista zahteva i “razbijanje” tih zahteva na manje celine – na ovaj način, developer lakše prave predikciju o vremenu potrebnom za izradu svakog zahteva zasebno.

Okvirno predviđanje vremena potrebnog za izradu softverskog proizvoda developeri i stejkholderi definišu uzevši u obzir:

1. Optimistično vreme
2. Verovatno vreme
3. Pesimistično vreme
4. Ispostavljeno vreme

Ovo znači da vreme izrade predstavlja varijablu koju nije uvek moguće precizno definisati, te se u obzir uzima da optimični vremenski okviri mogu biti ostvareni u 10-20% slučajeva, dok se verovatno vreme ostvaruje u 50% slučajeva, a pesimistične estimacije vode od 80% do 90% uspeha što dodatno potrđuje činjenicu da veliki broj projekata “probija rokove” zbog nedovoljno realne esitmacije vremenskog okvira.

## Primena liste rokova (Task Duration List)

U sledećoj tabeli možemo videti okvirno vreme izvršavanje svakog od zadatka u danima posebno:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zadatak | Optimistično vreme | Najverovatnije vreme | Pesimistično vreme | Ispostavljeno vreme |
| Prikupljanje zahteva | 2 | 3 | 4 |  |
| Kreiranje arhitekture sistema | 4 | 5 | 6 |  |
| Kreiranje baze podataka | 1 | 1 | 2 |  |
| Dodeljivanje zadataka | 1 | 1 | 1 |  |
| Implementacija backend-a | 45 | 50 | 55 |  |
| Dizajn korisničkih interfejsa | 3 | 5 | 6 |  |
| Implementacija KI (Web) | 60 | 65 | 70 |  |
| Implementacija KI (Mobile) | 60 | 65 | 70 |  |
| Integraciono tesitranje | 5 | 7 | 10 |  |
| Ispravka grešaka | 5 | 10 | 15 |  |
| Implementacija bezbednostnih mehanizama | 5 | 6 | 7 |  |
| Postavljanje aplikacije na server | 2 | 2 | 3 |  |
| Ukupno | 190 | 220 | 249 |  |

# Mrežni plan projekta (Project Network Diagram)

## Teoretska postavka

Nakon definisanja konkretnih zadataka i vremena potrebnog za njihovu implementaciju na određenom softverskom proizvodu, važno je isplanirati redosled faza razvoja samog softvera. Iz tog razloga je važno uzeti u obzir međusobne odnose faza razvoja i mere njihove medjuzavisnosti. Zbog toga je važno napraviti Mrežni plan projekta koji definiše prioritete u izradi funkcionalnosti, pogotovo u delovima projekata u kojima se faze razvoja naslanjaju jedna na drugu. Prvenstveno je važno srodne i paralelne zadatke grupisati kroz faze razvoja na takav način da svi zadaci koji od njih budu zavisili budu predviđeni za izradu u narednoj fazi razvoja softverskog proizvoda. Dakle, međusobno nezavisni zadaci se mogu obavljati paralelno od strane dvoje različitih ljudi, a svi zahtevi koji su u korelaciji sa njima se moraju obaviti u narednim fazama koje će zahtevati prethodno implementiranu osnovu za njihov nastanak i razvoj.

Mrežni plan projekta daje nam uvid i u srodne zahteve koji formiraju „linije života“ koje razvijaju različite funkcionalnosti zasebno, koje se na kraju povezuju u celovit gotov softverski proizvod. Sa tim u vezi, mrežni plan nam daje uvid i u „Kritičnu liniju života“ koja u samom planiranju faza nosi najveću važnost. To je najduža i najkompleksnija linija na dijagramu jednog projekta, od koje esencijalno zavisi ishod samog projekta, odnosno uspeh softverskog proizvoda. „Kritična linija života“ ne spaja nužno sve faze razvoja proizvoda, već ubraja sve neophodne komponente jednog funkcionalnog proizvoda, što znači da bez ove linije softversko rešenje ne može biti isporučeno. Ostale putanje jesu „Linije snabdevanja“ koje predstavljaju redunantni deo samog proizvoda, međutim imaju veliki značaj kako bi proizvod bio isporučen u celosti, i kako bi svi zahtevi stejkholdera bili adekvatno ispunjeni.

Mrežni plan takođe zavisi i od liste rokova, u jednakoj meri u kojoj zavisi od liste zadataka. Prilikom kreiranja mrežnog plana, suštinski je važno pronaći i grupisati sve neophodne funkcionalnosti kako bi Kritična linija bila što konkretnije definisana. U ove svrhe je na listi rokova potrebno obratiti posebnu pažnju na kolonu verovatnog vremena, kako bi najduža linija na projektu bila optimalno predviđena.

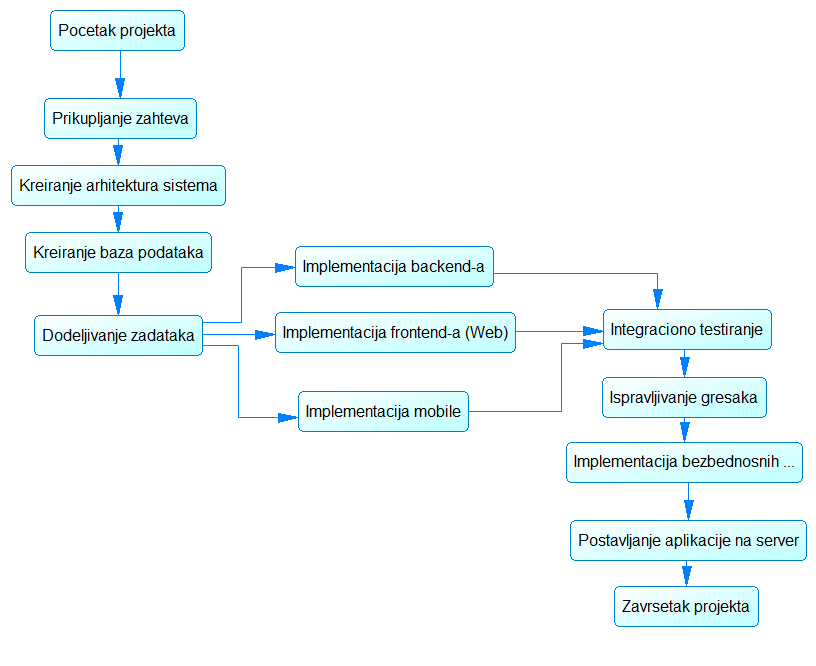
## Primena mrežnog projekta (Project Network Diagram)

Primenom mrežnog plana projekta na ovo mojoj aplikaciji daje nam na uvid koja komponenta zavisi od razvoja drugih, koje komponente se mogu paralelno razvijati kao i uvid u vremenski period realizacije komponenata.

Za početak nam je potrebno podsetiti se liste zadataka koje treba da obavimo.

**Lista zadataka koje potrebno izvršiti:**

1. Prikupljanje zahteva
2. Kreiranje arhitekture sistema
3. Kreiranje baze podataka
4. Dodeljivanje zadataka
5. Implementacija backend-a
6. Dizajn korisničkih interfejsa
7. Implementacija KI (Web)
8. Implementacija KI (Mobile)
9. Integraciono tesitranje
10. Ispravka grešaka
11. Implementacija bezbednostnih mehanizama
12. Postavljanje aplkacije na server



# Gantov dijagram raspored poslova (Grantt chart schedule)

## Teoretska postavka

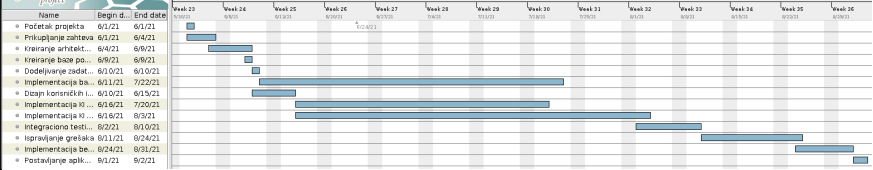
Ova tehnika se striktno fokusira na menadžment vremena kao važnog ograničavajućeg faktora i resursa. Osim predstavljanja podele poslova, važno je estimacijom odrediti dužinu trajanja faza projekata, u okviru kojih će se odvijati razvoj zactanih tokova razvoja funkcionalnosti jednog softverskog proizvoda.

Gantov dijagram, odnosno Gantogram, predstavlja grafički prikaz rokova sa analizom i podelama zaduženja u okviru jednog tima. Ovo znači da Gantov dijagram mora biti u koheziji sa svim ostalim tehnikama planiranja – listom zahteva, listom rokova, podelom zadataka itd. Ova tehnika nam pomaže da sve resurse rasporedimo optimalno u odnosu na konkretne vremenske rokove koji su nam omogućeni. Na taj način se sama estimacija potrebnog vremena precizira, te menadžerima ostaje dodatnog prostora i za vreme baferovanja, odnosno „kupovina dodatnog vremena“ koje se čuva za krajnje kvalitetnije testiranje proizvoda.

## Primena gantovog dijagrama rasporeda poslova (Grantt chart schedule)

Za izradu Gantovog dijagrama za razvoj ovog projekta, korišćen je “Gantt Project”.

Pre početka izrade Gantovog dijagrama moraju da se definišu liste poslova I rokova koja su definisana u predhodnim tehnikama. Zadaci se poklapaju za zadacima koje smo definisali, a za vreme je uzeta kolona za najverovatnije vreme.



# Lista budžeta projekta (Project Budget Chart)

## 

## Teoretska postavka

Odgovornost planiranja budžeta jednog projekta pripada menadžerima koji u skladu sa načinom raspolaganja resursima – ljudima, vremenom, prihodom – dolaze do optimalnog budžetiranja jednog projekta. Ukoliko utvrde da je projekat neisplativ, da neće povratiti uloženo uz dodatu vrednost, odbaciće projekat. Prilikom sastavljanja liste budžeta projekta, potrebno je voditi računa o sinergiji svih tehnika planiranja iz prethodnih faza planiranja projekta, a pogotovo je važno uzeti u obzir Gantov dijagram sa listom rokova.

Kao i vreme, tako i troškovi mogu biti optimistični, verovatni, pesimistični i ispostavljeni. Zbog toga se određenom formulom mora doći do „budžetske kontigencije“ koja podrazumeva krajnji ishod budžeta sa uračunatim faktorima koji na njega utiču. To znači da od svih zadataka oduzmemo polovinu pesimističnih i verovatnih troškova, te ovu sumu saberemo sa sumom najverovatnijih troškova. Tada dobijamo najpribližniju moguću vrednost izračunatu prema Goldratt aproksimaciji troškova.

## Primena liste budžete projekta (Project Budget Chart)

Pre predstavljanja budžeta moramo napraviti osvrt na tabelu projektovanog vremena za ispunjavanje zadataka.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zadatak | Optimistično vreme | Najverovatnije vreme | Pesimistično vreme | Ispostavljeno vreme |
| Prikupljanje zahteva | 2 | 3 | 4 |  |
| Kreiranje arhitekture sistema | 4 | 5 | 6 |  |
| Kreiranje baze podataka | 1 | 1 | 2 |  |
| Dodeljivanje zadataka | 1 | 1 | 1 |  |
| Implementacija backend-a | 45 | 50 | 55 |  |
| Dizajn korisničkih interfejsa | 3 | 5 | 6 |  |
| Implementacija KI (Web) | 60 | 65 | 70 |  |
| Implementacija KI (Mobile) | 60 | 65 | 70 |  |
| Integraciono tesitranje | 5 | 7 | 10 |  |
| Ispravka grešaka | 5 | 10 | 15 |  |
| Implementacija bezbednostnih mehanizama | 5 | 6 | 7 |  |
| Postavljanje aplikacije na server | 2 | 2 | 3 |  |
| Ukupno | 190 | 220 | 249 |  |

Sledeća tabela predstavlja troškove po zadatku, gde su cene predstavljene u evrima. Kao I kod sati izrade zadataka, cena je predstavljena optimistički, najverovatniji troškovi, pesimistički troškovi I ispostavljeni troškovi, gde će kolona za ispostavljene troškove ostati prazna kako smo još u fazi planiranja projekta. Kako bi se optimalno predvideo budžet projekta, optimalnu sumu budžeta računamo kao pesimistički troškovi minus najverovatniji troškovi pa sve do pomnožimo sa 0.5 i zatim saberemo sa najverovatnijim troškovima.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zadatak | Optimistični troškovi | Najverovatniji troškovi | Pesimistični troškovi | Ispostavljeni troškovi |
| Prikupljanje zahteva | 60 | 80 | 100 |  |
| Kreiranje arhitekture sistema | 60 | 80 | 100 |  |
| Kreiranje baze podataka | 20 | 20 | 40 |  |
| Dodeljivanje zadataka | 20 | 20 | 20 |  |
| Implementacija backend-a | 600 | 700 | 800 |  |
| Dizajn korisničkih interfejsa | 60 | 80 | 100 |  |
| Implementacija KI (Web) | 500 | 600 | 700 |  |
| Implementacija KI (Mobile) | 600 | 700 | 800 |  |
| Integraciono tesitranje | 100 | 140 | 200 |  |
| Ispravka grešaka | 100 | 200 | 300 |  |
| Implementacija bezbednostnih mehanizama | 100 | 120 | 140 |  |
| Postavljanje aplikacije na server | 40 | 40 | 60 |  |
| Ukupno | 2260 | 2780 | 3360 |  |

# Zaključak

Tehnike razvoja softvera su veoma bitne za razvoj nekog projekta. Ovde koristimo ove dve tehnike, listu uspešnosti projekta i struktura podele poslova WBS.

Na ovom projektom zadatku “pametna kuca” direktno smo promenili neke od tehnika planiranja projekta. Može se reći da su to jedne od najosnovnijih tehnika gde jasno u uvid imamo ko je zadužen za koj deo posla i koje zadatke obavlja, kao i koliko je potrebno vremena za obradu tih zadataka. Kako bi bili ozbiljni u očima svih koji očekuju od nas izradu projekta moramo ispoštovati ove tehnike planiranja, gde na kraju ovog dokumenta možemo reći da smo to i postigli.

Nakon što smo predhodno definisali zadatke i poslove, od jakog značaja je implementirati mrežni plan vođenja projekta kako bismo prezentovali tok izvršenja zadataka na samom projektu. Ovom tehnikom osim što možemo videti koja komponenta zavisi od koje, tj. koj zadatak zavisi od kog, vidimo okvirno vreme izvršavanja projekta, gde npr. Ukoliko smo u fazi implementacije funkcionalnosti pogledom na dijagram tačno možemo videti koje komponente su ostale za realizovati u narednom periodu. Time vodimo tačan plan izrade našeg ravoja samog proizvoda kojim se projekat bavi.

Gantov dijagram predstavlja jednu od ključnih tehnika plana vođenja projekta. Nakon ubacivanja definisanih zadataka, kao i dodeljivanja tih zadataka osobama u timu kroz definisanje vremenskog perioda izrade tih zadataka, u uvid dobijamo tačan pogled o tome koji su zadaci međusobno zavisni, koji su članovi tima opterećeni ili rasterećeni u kom periodu, kao I da li postoji neki slobdni prozor gde neko od članova time predugo čeka za izvršenje svog zadatka. Time možemo shvatiti da li smo negde pogrešili u dodeli i definisanju zadataka, kao i da li imamo nepotrebno preopterećenje ili rasterećenje članova tima.

Plan za budžet projekta predstavlja jedan od najkritičnijih aspekata planiranja projekta, jer tada stejkholderi i investitori imaju u uvid projektovanu cenu za izradu željenog sektora, pored toga što imaju u uvid tačne korake i zahteve za izradu projekta. Primer aplikacije za upravljanje rasporeda časova demonstrira budžet projekta koji je potreban za finansiranje tima. Nakon ovog dokumenta, možemo videti kako bi jedan plan budžeta projekta trebalo da izgleda.